The Densest k-Subhypergraph Problem

Eden Chlamtáč[*1], Michael Dinitz[†2], Christian Konrad[‡3], Guy Kortsarz[§4], and George Rabanca[¶5]

1 **Department of Computer Science, Ben Gurion University, Beersheva, Israel**
chlamtac@cs.bgu.ac.il.

2 **Dept. of Computer Science, Johns Hopkins University, Baltimore, MD, USA**
mdinitz@cs.jhu.edu

3 **ICE-TCS, School of Computer Science, Reykjavik University, Iceland**
christiank@ru.is

4 **Computer Science Department, Rutgers University, Camden, NY, USA**
guyk@crab.rutgers.edu

5 **Department of Computer Science, The Graduate Center, CUNY, USA**
grabanca@gradcenter.cuny.edu

Abstract

The Densest k-Subgraph (DkS) problem, and its corresponding minimization problem Smallest p-Edge Subgraph (SpES), have come to play a central role in approximation algorithms. This is due both to their practical importance, and their usefulness as a tool for solving and establishing approximation bounds for other problems. These two problems are not well understood, and it is widely believed that they do not an admit a subpolynomial approximation ratio (although the best known hardness results do not rule this out).

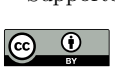
In this paper we generalize both DkS and SpES from graphs to hypergraphs. We consider the Densest k-Subhypergraph problem (given a hypergraph (V, E), find a subset $W \subseteq V$ of k vertices so as to maximize the number of hyperedges contained in W) and define the Minimum p-Union problem (given a hypergraph, choose p of the hyperedges so as to minimize the number of vertices in their union). We focus in particular on the case where all hyperedges have size 3, as this is the simplest non-graph setting. For this case we provide an $O(n^{4(4-\sqrt{3})/13+\epsilon}) \leq O(n^{0.697831+\epsilon})$-approximation (for arbitrary constant $\epsilon > 0$) for Densest k-Subhypergraph and an $\tilde{O}(n^{2/5})$-approximation for Minimum p-Union. We also give an $O(\sqrt{m})$-approximation for Minimum p-Union in general hypergraphs. Finally, we examine the interesting special case of interval hypergraphs (instances where the vertices are a subset of the natural numbers and the hyperedges are intervals of the line) and prove that both problems admit an exact polynomial time solution on these instances.

1998 ACM Subject Classification G.2.2 Graph Theory, F.2.2 Nonnumerical Algorithms and Problems

Keywords and phrases Hypergraphs, Approximation algorithms

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2016.6

[*] Partially supported by ISF grant 1002/14.
[†] Partially supported by NSF grants 1464239 and 1535887.
[‡] Supported by Icelandic Research Fund grants 120032011 and 152679-051.
[§] Partially supported by NSF grants 1218620 and 1540547.
[¶] Supported by ARL grant W911NF-09-2-0053



1 Introduction

Two of the most important outstanding problems in approximation algorithms are the approximability of the *Densest k-Subgraph* problem (DkS) and its minimization version, the *Smallest p-Edge Subgraph* problem (SpES or min-DkS). In DkS we are given as input a graph $G = (V, E)$ and an integer k, and the goal is to find a subset $V' \subseteq V$ with $|V'| = k$ which maximizes the number of edges in the subgraph of G induced by V'. In the minimization version, SpES, we are given a lower bound p on the number of required edges and the goal is to find a set $V' \subseteq V$ of minimum size so that the subgraph induced by V' has at least p edges. These problems have proved to be extremely useful: for example, a variant of DkS was recently used to obtain a new cryptographic system [3]. The same variant of the DkS problem was shown to be central in understanding financial derivatives [4]. The best-known algorithms for many other problems involve using an algorithm for Densest k-Subgraph or SpES as a black box (e.g. [22, 15, 11]).

Despite decades of work, very little is actually known about these problems. The first approximation ratio for DkS was $O(n^{2/5})$ [18] and was devised in 1993. These days, 23 years later, the best known ratio for the Densest k-Subgraph is $O(n^{1/4+\epsilon})$ for arbitrarily small constant $\epsilon > 0$ [7], and the best known approximation for SpES is $O(n^{3-2\sqrt{2}+\epsilon})$ for arbitrarily small constant $\epsilon > 0$ [9]. Given the slow improvement over 23 years, it is widely believed that DkS and SpES do not admit better than a polynomial approximation ratio. Furthermore, the existing approximation guarantees are tight assuming the recently conjectured hardness of finding a planted dense subgraph in a random graph (for certain parameters) [7, 9]. However, there has been very little progress towards an actual proof of hardness of approximation. It is clear that they are both NP-hard, but that is all that is known under the assumption that $P \neq NP$. Under much stronger complexity assumptions it is known that they cannot be approximated better than some constant [16, 12] or any constant [1], but this is still a long way from the conjectured polynomial hardness.

Based on the believed hardness of DkS and SpES, they have been used many times to give evidence for hardness of approximation. For example, consider the *Steiner k-Forest* problem in which the input is an edge weighted graph, a collection of q pairs $\{s_i, t_i\}_{i=1}^{q}$, and a number $k < q$. The goal is to find a minimum cost subgraph that connects at least k of the pairs. It is immediate to see that SpES is a special case of the Steiner k-forest problem[1], and hence it seems highly unlikely that the Steiner k-Forest problem admits a better than polynomial approximation ratios.

Given the interest in and importance of DkS and SpES, it is somewhat surprising that there has been very little exploration of the equivalent problems in *hypergraphs*. A hypergraph is most simply understood as a collection E of subsets over a universe V of vertices, where each $e \in E$ is called a *hyperedge* (so graphs are the special case when each $e \in E$ has cardinality 2). In general hypergraphs, the obvious extensions of DkS and SpES are quite intuitive. In the *Densest k-Subhypergraph* (DkSH) problem we are given a hypergraph (V, E) and a value k, and the goal is to find a set $W \subseteq V$ of size k that contains the largest number of hyperedges from E. In the *Minimum p-Union* (MpU) problem we are given a hypergraph and a number p, and the goal is to choose p of the hyperedges to minimize the size of their union.

Clearly these problems are at least as hard as the associated problems in graphs, but how much harder are they? Can we design nontrivial approximation algorithms? Can we extend

[1] Given an instance $(G = (V, E), p)$ of SpES, create an instance of Steiner k-Forest on a star with V as the leaves, uniform weights, a demand pair for each edge in E, and $k = p$.

the known algorithms for graphs to the hypergraph setting? Currently, essentially only lower bounds are known: Applebaum [2] showed that they are both hard to approximate to within n^ϵ for some fixed $\epsilon > 0$, assuming that a certain class of one-way functions exist. But it was left as an open problem to design any nontrivial upper bound (see footnote 5 of [2]).

1.1 Our Results

In this paper we provide the first nontrivial upper bounds for these problems. Let n denote the number of vertices and m denote the number of hyperedges in the input hypergraph. Our first result is an approximation for Minimum p-Union in general hypergraphs:

▶ **Theorem 1.** *There is an $O(\sqrt{m})$-approximation for the Minimum p-Union problem.*

We then switch our attention to the low rank case, since this is the setting closest to graphs. In particular, we focus on the 3-*uniform* case, where all hyperedges have size at most 3. In this setting it is relatively straightforward to design an $O(n)$-approximation for Densest k-Subhypergraph, although even this is not entirely trivial (the optimal solution could have size up to k^3 rather than k^2 as in graphs, which would make the trivial algorithm of choosing $k/3$ hyperedges only an $O(n^2)$-approximation rather than an $O(n)$-approximation as in graphs). We show that by very carefully combining a set of algorithms and considering the cases where they are all jointly tight we can significantly improve this approximation, obtaining the following theorem:

▶ **Theorem 2.** *For every constant $\epsilon > 0$, there is an $O(n^{4(4-\sqrt{3})/13+\epsilon}) \leq O(n^{0.697831+\epsilon})$-approximation for the Densest k-Subhypergraph problem on 3-uniform hypergraphs.*

Adapting these ideas to the minimization setting gives an improved bound for Minimum p-Union as well.

▶ **Theorem 3.** *There is an $\tilde{O}(n^{2/5})$-approximation for the Minimum p-Union problem on 3-uniform hypergraphs.*

It is worth noting that any f-approximation for DkSH can be used to give an $\tilde{O}(f)$-approximation for MpU (see Theorem 10), so Theorem 3 gives a significant improvement over this blackbox reduction from Theorem 2.

Finally, we define an interesting special case of Densest k-Subhypergraph and Minimum p-Union that can be solved exactly in polynomial time. Suppose we have an *interval hypergraph*: a hypergraph in which the vertices are a finite subset of $\mathbb{N}$ and each hyperedge is an interval of the real line (restricted to the vertices). Then we show that a dynamic programming algorithm can be used to actually solve our problems.

▶ **Theorem 4.** *Densest k-Subhypergraph and Minimum p-Union can be solved in polynomial time on interval hypergraphs.*

1.2 Related Work

As discussed, the motivation for these problems mostly comes from the associated graph problems, which have been extensively studied and yet are still poorly understood. The Densest k-Subgraph problem was introduced by Kortsarz and Peleg [18], who gave an $O(n^{2/5})$ ratio for the problem. Feige, Kortsarz and Peleg [13] improved the ratio to $O(n^{1/3-\epsilon})$ for ϵ that is roughly 1/60. The current best-known approximation for DkS is $O(n^{1/4+\epsilon})$ for arbitrarily small constant $\epsilon > 0$, due to Bhaskara et al. [7]. For many years the minimization

version, SpES, was not considered separately, and it was only relatively recently that the first separation was developed: building on the techniques of [7] but optimizing them for the minimization version, Chlamtáč, Dinitz, and Krauthgamer [9] gave an $O(n^{3-2\sqrt{2}+\epsilon})$-approximation for S$p$ES for arbitrarily small constant $\epsilon > 0$.

While defined slightly differently, DkSH and MpU were introduced earlier by Applebaum [2] in the context of cryptography: he showed that if certain one way functions exist (or that certain pseudorandom generators exist) then DkSH is hard to approximate within n^{ϵ} for some constant $\epsilon > 0$. Based on this result, DkSH and MpU were used to prove hardness for other problems, such as the k-route cut problem [10]. To the best of our knowledge, though, there has been no previous work on algorithms for these problems.

1.3 Organization

We begin in Section 2 with some preliminaries, showing the basic relationships between the problems. In Section 3 we give our $O(\sqrt{m})$-approximation for MpU in general hypergraphs. We then focus on small-rank hypergraphs, giving an $O(n^{4/5})$-approximation for DkSH on 3-uniform hypergraphs in Section 4, which we then improve to roughly $O(n^{0.698})$ in Section 5. We follow this in Section 6 with our improved bound for MpU on 3-uniform hypergraphs. Finally in Section 7 we show how to solve both problems exactly in polynomial time on interval hypergraphs. We conclude in Section 8 with some open questions for future work.

2 Preliminaries and Notation

A *hypergraph* $H = (V, E)$ consists of a set V (the vertices) together with a collection $E \subseteq 2^V$ (the hyperedges), where each hyperedge is a subset of V. We will typically use $n = |V|$ and $m = |E|$ to denote the number of vertices and hyperedges respectively. The *degree* of a vertex in a hypergraph is the number of hyperedges which contain it. Given a subset $V' \subseteq V$, the subhypergraph of H *induced* by V' is $H[V'] = (V', E_H)$ where $E_H = \{e \in E : e \subseteq V'\}$. We say that H is α-*uniform* if $|e| = \alpha$ for all $e \in E$, and that the *rank* of H is $\max_{e \in E} |e|$ (i.e. the smallest α such that all edges have cardinality at most α). A hyperedge e is *covered* by a set of vertices V' if $e \subseteq V'$.

Given a graph $G = (V, E)$ and a vertex $v \in V$, we use $\Gamma_G(v)$ to denote the set of nodes adjacent to v, and for a subset $V' \subseteq V$ we let $\Gamma_G(V') = \cup_{v \in V'} \Gamma(v)$. If G is clear from context, we will sometimes drop the subscript.

The main problems that we will consider are the following.

▶ **Definition 5.** Given a hypergraph $H = (V, E)$ and an integer k, the *Densest k-Subhypergraph* problem (DkSH) is to find a set $V' \subseteq V$, with $|V'| = k$, such that the number of edges in $H[V']$ is maximized.

▶ **Definition 6.** Given a hypergraph $H = (V, E)$ and an integer p, the *Minimum p-Union* problem (MpU) is to find a set $E' \subseteq E$, with $|E'| = p$, such that $|\cup_{e \in E'} e|$ is minimized.

Note that on 2-uniform hypergraphs, these two problems are the classic graph problems DkS and SpES respectively.

A special class of hypergraphs that we will consider are *interval hypergraphs*, defined as follows.

▶ **Definition 7.** $H = (V, E)$ is an *interval hypergraph* if V is a finite subset of $\mathbb{N}$ and for each $e \in E$ there are values $a_e, b_e \in \mathbb{N}$ such that $e = \{i \in V : a_e \leq i \leq b_e\}$.

2.1 Relationship Between Problems

We begin by proving some relatively straightforward relationships between the two problems. We first make the obvious observation that a solution for one problem implies a solution for the other.

▶ **Observation 8.** *If there exists a polynomial time algorithm that solves the Densest k-Subhypergraph problem for any k on a hypergraph H, then there exists a polynomial time algorithm that solves the Minimum p-Union problem on the hypergraph H. Similarly, if there is an algorithm that solves MpU on H, then there is an algorithm that solves DkSH on H.*

The relationship is not quite so simple when we are reduced to approximating the problems, but it is relatively straightforward to show that a relationship still exists. This is given by the following lemma, which will also prove to be useful later.

▶ **Lemma 9.** *If there exists an algorithm which in a hypergraph H containing a subhypergraph with k vertices and p hyperedges finds a subhypergraph (V', E') with $|V'| \leq fk$ and $|E'| \geq |V'|p/(kf)$, we can get an $O(f \log p)$-approximation for Min p-Union.*

Since any f-approximation algorithm for Densest k-Subhypergraph satisfies the conditions of the lemma, as an immediate corollary we get the following:

▶ **Theorem 10.** *If there is an f-approximation for Densest k-Subhypergraph, then there is an $O(f \log p)$-approximation for Minimum p-Union.*

Proof of Lemma 9. Let $(H = (V, E), p)$ be an instance of Minimum p-Union, and let $\mathcal{A}$ be an algorithm as described in the lemma. We assume without loss of generality that we know the number of nodes k in the optimal solution (since we can just try all possibilities for k), and hence that there exists a set $V^* \subseteq V$ with $|V^*| = k$ such that V^* covers at least p hyperedges. Initialize $E' = \emptyset$, and consider the following algorithm for Minimum p-Union that repeats the following until $|E'| \geq p$.

1. Let $V' = \mathcal{A}(H, k)$, and let E'' be the hyperedges of H covered by V'.
2. Let $E' \leftarrow E' \cup E''$.
3. Remove E'' from H (remove only the edges, not the corresponding vertices).

We claim that this is an $\tilde{O}(f)$-approximation for Minimum p-Union. Indeed, suppose at iteration i we added x_i vertices, and that at the beginning of the iteration, we had already added $p - p_i$ edges to the solution. In particular, that means that at least p_i of the original hyperedges contained in V^* were not yet removed. This then implies that the number of edges added in iteration i was at least $x_i \cdot p_i/(kf)$. Thus the number of edges we still need to add after iteration i is $p_{i+1} \leq p_i - x_i \cdot p_i/(kf) = p_i(1 - x_i/(kf))$. Thus by induction, after t iterations, the number of hyperedges we need to add is bounded by

$$p_{t+1} \leq p \prod_{i=1}^{t} (1 - x_i/(kf)) \leq p \exp\left(-\sum_{i=1}^{t} x_i/(kf)\right).$$

Thus, as soon as the total number of vertices added exceeds $kf \ln p$ for the first time, the number of edges will exceed p. Since the last iteration adds at most kf vertices, we are done. ◀

A standard argument also shows a (more lossy) reduction in the other direction.

▶ **Theorem 11.** *If there is an f-approximation for Minimum p-Union on α-uniform hypergraphs, then there is an $O(f^\alpha)$-approximation for Densest k-Subhypergraph on α-uniform hypergraphs (when $\alpha = O(1)$).*

Algorithm 1: $2\sqrt{m}$-approximation algorithm for the Minimum p-Union problem

Data: Bipartite input graph $G = (E, V, F)$ with $m = |E|$, $n = |V|$, parameter p

$E' \leftarrow \{\}$;
repeat
  $E'' \leftarrow$ Min-Exp$(G[E \setminus E', V])$;
  if $|E'| + |E''| \leq p$ **then**
    $E' \leftarrow E' \cup E''$;
  else
    Add arbitrary $p - |E'|$ nodes from E'' to E';
until $|E'| \geq p - \sqrt{m}$;
$E'' \leftarrow$ subset of $p - |E'|$ nodes of $E \setminus E'$ of smallest degree;
$E' \leftarrow E' \cup E''$;
return E';

3 Minimum p-Union in General Hypergraphs

Given a hypergraph $H = (V, E)$, in this section we work with the bipartite *incidence graph* $G = (E, V, F)$ of H, where $F = \{(e, v) \in E \times V : v \in e\}$. Solving M$p$U on H corresponds to finding a subset $E' \subseteq E$ of p vertices in G of minimum vertex expansion, i.e., E' such that $|\Gamma_G(E')|$ is minimized.

Our algorithm requires a subroutine that returns a subset of vertices of minimum expansion (without the cardinality bound on the set). In other words, we need a polynomial-time algorithm Min-Exp(G) which returns a subset of E so that

$$\frac{|\text{Min-Exp}(G)|}{|\Gamma_G(\text{Min-Exp}(G))|} \geq \frac{|E'|}{|\Gamma_G(E')|},$$

for every subset $E' \subseteq E$.

Minimally expanding subsets of this kind have previously been used (e.g. in [17, 14]) in communication settings where computation time is disregarded, but in our context we need a polynomial-time algorithm. In Appendices A and B we give two different algorithms for doing this. The first, in Appendix A, uses a reduction to network flows. The second, in Appendix B, is based on a straightforward adaptation of a linear programming approach for the graph case due to Charikar [8]. In order to simplify the presentation, we will for the rest of the section assume that we have such an algorithm and will defer them to the appendices.

In the following, for subsets $E' \subseteq E$ and $V' \subseteq V$, we denote the induced subgraph of G by vertex set $E' \cup V'$ by $G[E', V']$.

In the first phase, our algorithm (Algorithm 1) iteratively adds vertices E'' to an initially empty set E' until E' exceeds the size $p - \sqrt{m}$. The set E'' is a minimally expanding subset in the induced subgraph $G[E \setminus E', V]$. If E'' is large so that $|E' \cup E''| > p$, then an arbitrary subset of E'' is added to E' so that E' has the desired size p. Then, in the second phase, we add the $p - |E'|$ vertices of $E \setminus E'$ of smallest degree to E' (ties broken arbitrarily), and the algorithm returns set E'.

▶ **Theorem 12.** *Algorithm 1 is a $(2\sqrt{m})$-approximation algorithm for MpU.*

Proof. Let $OPT \subseteq E$ be an optimal solution and let $r = |\Gamma_G(OPT)|$. Let E'_i denote the set E' in the beginning of the ith iteration of the repeat loop. Suppose that the algorithm runs

in l rounds. Then, E'_{l+1} is the set E' after the last iteration of the loop, but before the nodes selected in Line 9 are added.

Consider an arbitrary iteration $i \leq l$ and let $E'' \leftarrow \text{Min-Exp}(G[E \setminus E'_i, V])$ as in the algorithm. Note that by the condition of the loop, we have $|E'_i| \leq p - \sqrt{m}$. Furthermore, we have

$$\frac{|E''|}{|\Gamma_G(E'')|} \geq \frac{|OPT \setminus E'_i|}{|\Gamma_G(|OPT \setminus E'_i|)|} \geq \frac{p - |E'_i|}{r},$$

since E'' is a set of minimum expansion. Then,

$$|\Gamma_G(E'')| \leq \frac{|E''|r}{p - |E'_i|} \leq \frac{|E''|r}{p - p + \sqrt{m}} = \frac{|E''|r}{\sqrt{m}}.$$

Thus, we have $|\Gamma_G(E'_{i+1})| \leq |\Gamma_G(E'_i)| + \frac{|E''|r}{\sqrt{m}}$ (note that this inequality also captures the case when only a subset of E'' is added to E' in Line 7). Now, note that the sets E'' of any two different iterations are disjoint and thus the sizes of the sets E'' of the different iterations sum up to at most m. We thus obtain the bound:

$$|\Gamma_G(E'_{l+1})| \leq \frac{mr}{\sqrt{m}} = \sqrt{m}r.$$

In phase two, we select at most $\sqrt{m}$ vertices E'' of minimum degree in $G[E \setminus E', V]$. Clearly, the maximum degree of these vertices is at most r (if it was larger, then $|\Gamma_G(OPT)|$ would be larger as well) and thus $|\Gamma_G(E'')| \leq \sqrt{m}r$. The neighborhood of the returned set of our algorithm is hence at most $2\sqrt{m}r$ which gives an approximation factor of $2\sqrt{m}$. ◀

4 Densest k-Subhypergraph in 3-uniform hypergraphs

In this section, we consider the Densest k-Subhypergraph problem in 3-uniform hypergraphs. We develop an $O(n^{4/5})$-approximation algorithm here, and show in Section 5 how to improve the approximation factor to $O(n^{0.697831+\epsilon})$, for any $\epsilon > 0$, by replacing one of our subroutines with an algorithm of Bhaskara et al. [7].

Throughout this section, let $H = (V, E)$ be the input 3-uniform hypergraph. Let $K \subseteq V$ denote an optimal solution, i.e., a subset of vertices such that $H[K]$ is a densest k-subhypergraph. The average degree of $H[K]$ is denoted by $d = 3|E(H[K])|/k$. We say that a hyperedge is optimal if it is contained in $H[K]$.

4.1 Overview of our Algorithm

Let $K_1 \subseteq V$ be a set of $k/3$ vertices of largest degree (ties broken arbitrarily), Δ the minimum degree of a node in K_1, and $H' = H[V \setminus K_1]$. Note that the maximum degree in H' is Δ.

Suppose first that at least half of the optimal hyperedges contain at least one vertex of K_1. Then the following lemma shows that we can easily achieve a much better approximation than we are aiming for:

▶ **Lemma 13.** *Suppose that at least half of the optimal hyperedges contain a vertex of K_1. Then we can achieve an $O(n^{1/4+\varepsilon})$ approximation for any $\varepsilon > 0$.*

Proof. By our assumption, there is a set P of optimal hyperedges of size at least $dk/6$ such that every edge in P intersects K_1. Consider two cases.

Case 1: For at least half the edges $e \in P$, we have $|e \cap K_1| \geq 2$. Denote the set of these edges by P'. For every vertex $u \in V$, let its K_1-weight be the number of pairs $\{v, x\}$ such

Algorithm 2: Greedy algorithm for Densest k-Subhypergraph in 3-uniform hypergraphs

Data: 3-uniform Hypergraph $H = (V, E)$, parameter k, vertex set $K_1 \subseteq V$ of size $k/3$

1 For every $v \in V$, let its K_1-degree be $|\{e \in E \mid v \in e, e \cap K_1 \neq \emptyset\}|$;
2 $K_2 \leftarrow$ a set of $k/3$ vertices of highest K_1-degree (K_1 and K_2 may intersect);
3 For any $u \in V$, let its (K_1, K_2)-degree be the number of edges of the form $(u, v, x) \in E$ such that $v \in K_2$ and $x \in K_1$;
4 $K_3 \leftarrow$ a set of $k/3$ vertices of highest (K_1, K_2)-degree. (K_3 may intersect K_1 and/or K_2);
5 **return** $K_1 \cup K_2 \cup K_3$;

$v, x \in K_1$ and $\{u, v, x\}$ is a hyperedge. Then by our assumption, the vertices in K have average K_1-weight at least $|P'|/k \geq d/12$. Choosing $2k/3$ vertices greedily (by maximum K_1-weight) gives (along with K_1) a k-subhypergraph with at least $dk/18$ hyperedges.

Case 2: $P'' = P \setminus P'$ contains at least half the hyperedges in P. Note that $|e \cap K_1| = 1$ for every $e \in P''$. For every pair of vertices $u, v \in V \setminus K_1$, let its K_1-weight be the number of vertices $x \in K_1$ such that $\{u, v, x\}$ is a hyperedge, and let G be the graph on vertices $V \setminus K_1$ with these edge weights. Then any k'-subgraph of G with total edge weight w corresponds to a $(|K_1| + k')$-subhypergraph of H with at least w hyperedges, and in particular, G contains a k-subgraph with average weighted degree at least $2|P''|/k \geq d/6$, which can be easily pruned (randomly or greedily) down to a $2k/3$-subgraph with average weighted degree $\Omega(d)$. Thus we can run the Densest k-Subgraph approximation algorithm of Bhaskara et al. [7][2], and find a $2k/3$-subgraph of G with total weight at least $kd/n^{1/4+\varepsilon}$, which in turn gives a $(|K_1| + 2k/3 =)k$-subhypergraph of H with a corresponding number of hyperedges. ◀

In the more difficult case, at least half of the optimal hyperedges are fully contained in H'. Exploiting the fact that the maximum degree in H' is Δ and trading off multiple algorithms, we show in the following subsection how to obtain an $O(n^{\frac{4}{5}})$-approximation algorithm in this case.

4.2 An $O(n^{4/5})$-approximation

We start with a greedy algorithm similar to the greedy algorithm commonly used for Densest k-Subgraph [18, 13, 7].

Algorithm 2 selects a subset K_2 of $k/3$ vertices v with largest K_1-degree, i.e., the number of hyperedges incident to v that contain at least one vertex of K_1. Then, a subset K_3 of $k/3$ vertices w with largest (K_1, K_2)-degree is selected, where the (K_1, K_2)-degree of w is the number of hyperedges containing w of the form $\{w, x, y\}$ with $x \in K_1$ and $y \in K_2$. Note that the sets K_1, K_2 and K_3 are not necessarily disjoint and the returned set may thus be smaller than k.

The following lemma gives a lower bound on the average degree guaranteed by this algorithm. It is a straightforward extension of similar algorithms for graphs.

▶ **Lemma 14.** *Algorithm 2 returns a k-subhypergraph with average degree $\Omega(\Delta k^2/n^2)$.*

[2] Strictly speaking, the algorithm in [7] is defined for unweighted graphs, but one can easily adapt it by partitioning the edges into $O(\log n)$ sets with similar edge weights, and running the algorithm separately on every set of edges, thus losing only an additional $O(\log n)$ factor in the approximation.

Algorithm 3: A neighborhood-based algorithm for Densest k-Subhypergraph in 3-uniform hypergraphs

Data: 3-uniform Hypergraph $H' = (V', E')$ and parameter k.

foreach *vertex* $v \in V$ **do**
  $G_v \leftarrow (V \setminus \{v\}, \{(u,x) \mid (v,u,x) \in E\})$;
  foreach *integer* $\hat{d} \in [k-1]$ **do**
    $G_v^{\hat{d}} \leftarrow G_v$;
    while *there exists a vertex* u *in* $G_v^{\hat{d}}$ *of degree* $< \hat{d}$ **do**
      delete u from $G_v^{\hat{d}}$;
    $S_v^{\hat{d}} \leftarrow$ a set of $(k-1)/2$ vertices with highest degree in $G_v^{\hat{d}}$;
    $T_v^{\hat{d}} \leftarrow$ a set of $(k-1)/2$ vertices with the most neighbors in $S_v^{\hat{d}}$;
return *The densest among all subhypergraphs* $H'[\{v\} \cup S_v^{\hat{d}} \cup T_v^{\hat{d}}]$ over all choices of $v, \hat{d}$;

Proof. By choice of K_1 and definition of Δ, every vertex in K_1 has degree at least Δ, and so the total number of edges containing vertices in K_1 is at least $\Delta|K_1|/3 = \Delta k/9$ (since we could potentially be double-counting or triple-counting some edges).

If we were to choose n vertices for K_2, there would be at least $\Delta k/9$ edges containing both a vertex in K_1 and a vertex in K_2 (as noted above). Choosing $k/3$ vertices greedily out of n yields a set K_2 such that there are at least $\Delta k/9 \cdot (k/3)/n = \Delta k^2/(27n)$ such edges.

Finally, choosing the $k/3$ vertices with the largest contribution (out of n) for K_3 ensures that there will be at least $\Delta k^2/(27n) \cdot (k/3)/n = \Omega(\Delta k^3/n^2)$ edges in $E \cap K_1 \times K_2 \times K_3$, giving average degree $\Omega(\Delta k^2/n^2)$. ◀

We now offer a second algorithm, which acts on H' and is based on neighborhoods of vertices.

Algorithm 3 exploits the bound on the maximum degree in H' to find a dense hypergraph inside the neighborhood of any vertex of degree $\Omega(d)$ in K, by considering the neighborhood of a vertex as a graph. Pruning low-degree vertices in this graph (which would not contribute many hyperedges to K) helps reduce the size of the graph, and makes it easier to find a slightly denser subgraph. Since the vertices of K and their degrees are not known, the algorithm tries all possible vertices.

▶ **Lemma 15.** *If H' contains a k-subhypergraph with average degree $d' = \Omega(d)$, then Algorithm 3 returns a k-subhypergraph with average degree $\Omega(d^2/(\Delta k))$.*

Proof. Since at the end of the algorithm we take the densest induced subhypergraph of H' (among the various choices), it suffices to show that there is some choice of v and $\hat{d}$ which gives this guarantee. So let v be an arbitrary vertex in K with degree (in K) at least d'. We know that G_v contains a subgraph with at most k vertices and at least d' edges, so its average degree is at least $2d'/k$. Setting $\hat{d} = d'/(2k)$, we know that the pruning procedure can remove at most $k \cdot d'/(2k) = d'/2$ out of the d' edges in this subgraph, so the subgraph still retains at least $d'/2$ edges. On the other hand, we know that G_v has at most Δ edges (since we've assumed the maximum degree in H' is at most Δ), and therefore, the same holds for the graph $G_v^{\hat{d}}$, in which the minimum degree is now at least $d'/2k$. This means that $G_v^{\hat{d}}$ has at most $2\Delta/(d'/2k) = O(\Delta k/d)$ vertices.

Since there exists a k-subgraph of $G_v^{\hat{d}}$ with $\Omega(d)$ edges, the greedy choice of $S_v^{\hat{d}}$ must give some set in which at least $\Omega(d)$ edges are incident. The greedy choice of $T_v^{\hat{d}}$ then reduces the

lower bound on the number of edges by a $((k-1)/2)/|V(G_v^{\hat{d}})| = \Omega(d/\Delta)$ factor, giving us $\Omega(d^2/\Delta)$ edges. However, by the definition of G_v, together with v these edges correspond to hyperedges in H'. Thus, the algorithm returns a k-subhypergraph with $\Omega(d^2/\Delta)$ hyperedges, or average degree $d^2/(\Delta k)$. ◀

Combining the various algorithms we've seen with a trivial algorithm and choosing the best one gives us the following guarantee:

▶ **Theorem 16.** *There is an $O(n^{4/5})$-approximation for Dense k-Subhypergraph in 3-uniform hypergraphs.*

Proof. By Lemma 13, if at least half the optimal edges intersect K_1, then we can achieve a significantly better approximation (namely, $n^{1/4+\varepsilon}$). Thus, from now on let us assume this is not the case. That is, H' still contains a k-subhypergraph with average degree $\Omega(d)$. Again, recall that the maximum degree in H' is at most Δ.

By Lemma 14, Algorithm 2 gives us a k-subhypergraph with average degree $d_1 = \Omega(\Delta k^2/n^2)$. On the other hand, applying Algorithm 3 to H' will give us a k-subhypergraph with average degree $d_2 = \Omega(d^2/(\Delta k))$ by Lemma 15.

Finally, we could choose $k/3$ arbitrary edges in H and the subhypergraph induced on the vertices they span, giving us average degree $d_3 \geq 1$. Thus, the best of the three will give us a k-subhypergraph with average degree at least

$$\max\{d_1, d_2, d_3\} \geq (d_1^2 d_2^2 d_3)^{1/5} = \Omega((\Delta^2 k^4/n^4 \cdot d^4/(\Delta^2 k^2))^{1/5}) = d \cdot \Omega((k^2/d)^{1/5}/n^{4/5}).$$

Since we must have $k^2/d \geq 1$, the above gives an $O(n^{4/5})$-approximation. ◀

5 An improved approximation for 3-uniform Densest k-Subhypergraph

In Section 4 we gave an $O(n^{4/5})$ approximation which combined a greedy algorithm with Algorithm 3, which looked for a dense subgraph inside a graph defined by the neighborhood of a vertex in H. To find this dense subgraph, we used a very simple greedy approach. However, we have at our disposal more sophisticated algorithms, such as that of Bhaskara et al. [7]. One way to state the result in that paper (see Bhaskara's PhD thesis for details on this version [6]) is as follows:

▶ **Theorem 17.** *In any n-vertex graph G, for any $\alpha \in [0,1]$, if $k = n^\alpha$, then Densest k-Subgraph in G can be approximated within an $n^\varepsilon k^{1-\alpha}$ factor in time $n^{O(1/\varepsilon)}$ for any $\varepsilon > 0$.*

The $n^{1/4+\varepsilon}$ guarantee of [7] follows since for any $\alpha \in [0,1]$, we have $k^{1-\alpha} = n^{\alpha(1-\alpha)} \leq n^{1/4}$.

Using this guarantee instead of the simple greedy algorithm for DkS, we get the following improved algorithm for 3-uniform Densest k-Subhypergraph:

The approximation guarantee in this final algorithm is given by the following lemma:

▶ **Lemma 18.** *Let H' be an n-vertex 3-uniform hypergraph with maximum degree $\leq \Delta$, containing a k-subhypergraph of average degree d', and let α, β be such that $k = n^\alpha$ and $\Delta k/d' = n^\beta$. Then Algorithm 4 returns a k-subhypergraph of H of average degree*

$$\Omega\left(\frac{d'}{n^{\varepsilon+\alpha(2-\alpha/\min\{\beta,1\})}}\right).$$

Algorithm 4: A DkS-based algorithm for Densest k-Subhypergraph in 3-uniform hypergraphs

Data: 3-uniform Hypergraph $H' = (V', E')$ and parameters k and $\varepsilon > 0$.

foreach *vertex* $v \in V$ **do**
  $G_v \leftarrow (V \setminus \{v\}, \{(u,x) \mid (v,u,v) \in E\})$;
  foreach *integer* $\hat{d} \in [k-1]$ **do**
    $G_v^{\hat{d}} \leftarrow G_v$;
    while *there exists a vertex* u *in* $G_v^{\hat{d}}$ *of degree* $< \hat{d}$ **do**
      Delete u from $G_v^{\hat{d}}$;
    $K_v^{\hat{d}} \leftarrow$ the vertex set returned by the algorithm of Bhaskara et al. [7] on the graph $G_v^{\hat{d}}$ with parameters $k-1$ and ε;
return *The densest among all subhypergraphs* $H'[\{v\} \cup K_v^{\hat{d}}]$ over all choices of $v, \hat{d}$;

Proof. As in the proof of Lemma 15, we can deduce that for at least some choice of v and $\hat{d}$, the graph $G_v^{\hat{d}}$ has at most $\min\{n, O(\Delta k/d')\} = O(n^{\min\{1,\beta\}})$ vertices and contains a k-subgraph with average degree $\Omega(d'/k)$.

By Theorem 17, since $k = n^\alpha = \Omega(|V(G_v^{\hat{d}})|^{\alpha/\min\{1,\beta\}})$, the algorithm of [7] will return a $(k-1)$-subgraph of $G_v^{\hat{d}}$ with average degree

$$\Omega\left(\frac{d'/k}{n^\varepsilon k^{1-\alpha/\min\{\beta,1\}}}\right) = \Omega\left(\frac{d'}{n^{\varepsilon+\alpha(2-\alpha/\min\{\beta,1\})}}\right).$$

As noted in the proof of Lemma 15, this corresponds to a k-subhypergraph of H' with the same guarantee. ◀

▶ Remark. In the notation of Lemma 18 we have $\Delta/d' = n^{\beta-\alpha}$ which implies that $\beta \geq \alpha$ (since $\Delta \geq d'$).

Trading off the various algorithms we have seen, we can now prove the guarantee stated in Theorem 2.

▶ **Theorem 19** (Theorem 2 restated). *For every constant $\varepsilon > 0$, there exists a polynomial time algorithm that achieves an $O(n^{4(4-\sqrt{3})/13+\varepsilon}) \leq O(n^{0.697831+\varepsilon})$-approximation for Densest k-Subhypergraph in 3-uniform hypergraphs.*

Proof. By Lemma 13, if at least half the optimal edges intersect K_1, then we can achieve a significantly better approximation (namely, $n^{1/4+\varepsilon}$). Thus, from now on let us assume this is not the case. That is, H' still contains a k-subhypergraph with average degree $\Omega(d)$. Again, recall that the maximum degree in H' is at most Δ.

As before, let α, β be such that $k = n^\alpha$ and $\Delta k/d = n^\beta$. By Lemma 14, Algorithm 2 gives us a k-subhypergraph with average degree

$$d_1 = \Omega(\Delta k^2/n^2) = \Omega\left(\frac{d}{(d/\Delta)n^2/k^2}\right) = \Omega\left(\frac{d}{n^{\alpha-\beta}n^{2-2\alpha}}\right) = \Omega\left(\frac{d}{n^{2-\alpha-\beta}}\right).$$

On the other hand, by Lemma 18, Algorithm 4 to H' will give us a k-subhypergraph with average degree

$$d_2 = \Omega\left(\frac{d}{n^{\varepsilon+\alpha(2-\alpha/\min\{\beta,1\})}}\right).$$

Let us analyze the guarantee given by the best of Algorithm 2 and Algorithm 4. First, consider the case of $\beta > 1$. In this case, taking the best of the two gives us approximation ratio at most $n^{\varepsilon+\min\{2-\alpha-\beta,\alpha(2-\alpha)\}} \leq n^{\varepsilon+\min\{1-\alpha,\alpha(2-\alpha)\}}$. It is easy to check that this minimum is maximized when $\alpha = (3-\sqrt{5})/2$ giving approximation ratio $n^{(\sqrt{5}-1)/2+\varepsilon} \leq n^{0.618034+\varepsilon}$, which is even better than our claim.

Now suppose $\beta \leq 1$. In this case, the approximation guarantee is $n^{\varepsilon+\min\{h_1,h_2\}}$, where $h_1 = 2-\alpha-\beta$ and $h_2 = \alpha(2-\alpha/\beta)$. If $\alpha \geq 2/3$, then it can be checked that we always have $h_1 \leq h_2$ for any $\beta \in [\alpha, 1]$, in which case we have approximation factor at most $n^{\varepsilon+2-2/3-2/3} = n^{2/3+\varepsilon}$, which is again better than our claim. On the other hand, if $\alpha \leq (3-\sqrt{5})/2$, then $h_2 \leq h_1$ for any $\beta \leq 1$, and so for this range of α we get approximation factor at most $n^{\varepsilon+\alpha(2-\alpha)} \leq n^{(\sqrt{5}-1)/2}$, which as we've noted is also better than our claim. Finally, if $\alpha \in ((3-\sqrt{5})/2, 2/3)$ then a straightforward calculation shows that

$$\min\{h_1,h_2\} = \begin{cases} h_1 & \text{if } \beta \geq 1 - \frac{3\alpha}{2} + \sqrt{1-3\alpha+13\alpha^2/4} \\ h_2 & \text{otherwise,} \end{cases}$$

and that the value of $\min\{h_1,h_2\}$ is maximized at this threshold value of β. And so for α in this range we have $\min\{h_1,h_2\} \leq 1+\alpha/2-\sqrt{1-3\alpha+13\alpha^2/4}$, which is maximized at $\alpha = \frac{18+2\sqrt{3}}{39} \approx 0.55$, giving approximation ratio $n^{\varepsilon+4(4-\sqrt{3})/13}$. ◀

6 Minimum p-Union in 3-uniform hypergraphs

In this section we explore Minimum p-Union (the minimization version of Densest k-Subhypergraph), and give the following guarantee:

▶ **Theorem 20.** *There is an $\tilde{O}(n^{2/5})$-approximation algorithm for Minimum p-Union in 3-uniform hypergraphs.*

Note that this is significantly better than the $n^{0.69\cdots}$-approximation we would get by reducing the problem to Densest k-Subhypergraph via Theorem 10 and applying the approximation algorithm from Theorem 2.

In this problem, we are given a 3-uniform hypergraph $H = (V, E)$, and a parameter p, the number of hyperedges that we want to find. Let us assume that the optimal solution, $P \subseteq E$, has k vertices (i.e. $|\cup_{e\in P} e| = k$). We do not know k, but the algorithm can try every possible value of $k = 1, \ldots, n$, and output the best solution. Thus, we assume that k is known, in which case the average degree in the optimum solution is $d = 3p/k$.

Recall that it is not necessary to get p edges in one shot. By Lemma 9, it is enough to find any subhypergraph of size at most $kn^{2/5}$ with average degree at least $\Omega(d/n^{2/5})$.

We follow along the lines of DkSH by choosing vertex set K_1 to be the $kn^{2/5}$ vertices of largest degree. The following lemma (corresponding to Lemma 13 for DkSH) shows that if at least half the edges in P intersect K_1, then by Lemma 9 we are done.

▶ **Lemma 21.** *Suppose that at least half of the optimal edges contain a vertex of K_1. Then we can find a subhypergraph with at most $O(kn^{2/5})$ vertices and average degree at least $\Omega(d/n^{2/5})$.*

Proof. By our assumption, there is a set of optimal hyperedges $P' \subset P$ of size at least $dk/6$ such that every edge in P' intersects K_1.

As in the proof of Lemma 13, if at least half the edges in P' intersect K_1 in more than one vertex, then we can easily recover a set of k vertices which along with K_1 contain at

least $\Omega(p) = \Omega(kd)$ hyperedges. Since $|K_1| = kn^{2/5}$, this subgraph has $O(kn^{2/5})$ vertices and average degree $\Omega(d/n^{2/5})$ as required.

Thus, we may assume that at least half the edges in P' intersect K_1 in exactly one vertex. Then again as in Lemma 13, we define a graph G on vertices $V \setminus K_1$ where every pair of vertices $u, v \in V \setminus K_1$ is an edge with weight $|\{x \in K_1 \mid (u, v, x) \in E\}|$. Once again, subgraphs of G with total edge weight w correspond to a subhypergraphs of H with at least w edges, and in particular, G contains a k-subgraph with average weighted degree at least $\Omega(d)$. Thus running the SpES approximation of [9] (or more precisely, the weighted version [11]), gives a subgraph with at most kf vertices and total edge weight at least $\Omega(kd)$ for some $f = n^{0.17+\varepsilon}$ (which is well below $n^{2/5}$). Once again, the corresponding subhypergraph has at most $|K_1| + kf = O(kn^{2/5})$ vertices, and so the average degree is at least $\Omega(d/n^{2/5})$ as required. ◀

Thus, we will assume from now on that at least half of the hyperedges in P *do not* contain at least one vertex from K_1, i.e. that $H' = H[V \setminus K_1]$ still contains at least half the hyperedges in P.

As with DkSH, we now proceed with a greedy algorithm. Starting with the same vertex set K_1 defined above, it follows from Lemma 14 that if we run Algorithm 2 on H with parameter $n^{2/5}k$, then we get a subhypergraph on $O(kn^{2/5})$ vertices induced on sets K_1, K_2, K_3 such that if the minimum degree in K_1 (which bounds the maximum degree in $V \setminus K_1$) is Δ, then the subhypergraph has average degree $\Omega(\Delta k^2 n^{4/5}/n^2)$. The total number of hyperedges in this subhypergraph is $\Omega(\Delta k^3 n^{6/5}/n^2) = \Omega(\Delta k^3/n^{4/5})$. If this is at least $p = dk/3$, then we are done. Thus, we will assume from now on that $\Delta k^3/n^{4/5} = O(dk)$, that is

$$\Delta = O\left(\frac{dn^{4/5}}{k^2}\right). \tag{1}$$

We reuse Algorithm 3 on H', which gives us the following guarantee:

▶ **Lemma 22.** *Applying Algorithm 3 to the above hypergraph H' with parameter*

$$\hat{k} = \frac{k\sqrt{p\Delta}}{d} = \sqrt{\frac{k^3\Delta}{3d}}$$

returns a subhypergraph with at most kf vertices and average degree at least d/f for some

$$f = O(\max\{k, n^{2/5}/\sqrt{k}\}).$$

Proof. As in the proof of Lemma 15, we can deduce that for at least some choice of v and $\hat{d}$, the graph $G_v^{\hat{d}}$ has at most $O(\Delta k/d)$ vertices and has minimum degree at least $\Omega(d/k)$.

Note that we may not even have $\hat{k}$ vertices in $G_v^{\hat{d}}$. If we do have at least $\hat{k}$ vertices, then the greedy choice of $S_v^{\hat{d}}$ gives us $\Omega(\hat{k}d/k)$ edges incident in the set (in fact, any choice of $\Omega(\hat{k})$ vertices would do). The greedy choice of $T_v^{\hat{d}}$ then reduces the number of edges by (in the worst case) a $\hat{k}/(\Delta k/d)$-factor, giving us a total number of edges

$$\Omega\left(\frac{\hat{k}^2 d^2}{\Delta k^2}\right) = \Omega(p).$$

Thus, in this case, we only need to bound the size of the subgraph. By (1), we can bound $\hat{k}$ as follows:

$$\hat{k} = \sqrt{\frac{k^3\Delta}{3d}} = O\left(\sqrt{\frac{dn^{4/5}}{k^2} \cdot \frac{k^3}{d}}\right) = O\left(k \cdot \frac{n^{2/5}}{\sqrt{k}}\right),$$

which proves the lemma for this case.

If we do not have $\hat{k}$ vertices in $G_v^{\hat{d}}$, then the algorithm simply returns $G_v^{\hat{d}}$ itself, which has at most $\hat{k} = O(k \cdot n^{2/5}/\sqrt{k})$ vertices and average degree at least $\Omega(d/k)$, as required.

As noted in the proof of Lemma 15, this corresponds to a subhypergraph of H' with the same guarantee. ◀

We can now prove the main theorem.

Proof of Theorem 20. By Lemma 22 and Lemma 9, to prove the theorem it suffices to show that $\max\{k, n^{2/5}/\sqrt{k}\} = O(n^{2/5})$. Since clearly $n^{2/5}/\sqrt{k} \leq n^{2/5}$, let us consider the parameter k. By definition of d and Δ, we clearly have $d \leq \Delta$, thus, by (1) we have

$$d \leq \Delta = O\left(\frac{dn^{4/5}}{k^2}\right)$$

which implies $k = O(n^{2/5})$, and so the theorem follows. ◀

7 Interval Hypergraphs

We show now that DkS and MpU can be solved in polynomial time on interval hypergraphs. We only give an algorithm for MpU; a similar algorithm for DkS follows then from Observation 8.

As defined in Section 2, a hypergraph $H = (V, E)$ is an interval hypergraph, if $V \subseteq \mathbb{N}$ and for each $e \in E$ there are integers a_e, b_e such that $e = \{i \in V : a_e \leq i \leq b_e\}$. Solving M$p$U on H can be interpreted as finding p intervals with minimum joint support.

▶ **Theorem 23.** Minimum p-Union *is solvable in polynomial time on interval hypergraphs.*

Proof. Let $b_1, ..., b_m$ be the largest elements in hyperedges $e_1, ..., e_m$ respectively, and assume that $b_i \leq b_j$ for any $i < j$. Similarly let $a_1, ..., a_m$ be the smallest elements in $e_1, ..., e_m$ respectively.

We present a dynamic programming algorithm which calculates for each $j \leq i$ the optimal solution to an instance of Minimum p-Union on the hyperedges $e_1, ..., e_i$ with $p = j$ under the constraint that e_i belongs to the solution. Let $A[i, j]$ store the value of this optimal solution. Assume that the values of A have been computed for all i', j' with $j' \leq i' < i$. We show how to compute $A[i, j]$ for any $j \leq i$.

We partition the hyperedges $e_1, ..., e_i$ in three sets A_i, B_i, C_i with A_i containing all hyperedges disjoint from e_i, B_i containing all hyperedges intersecting but not included in e_i, and C_i containing e_i and all hyperedges included in e_i (see Fig. 1). Therefore we have:

1. $b_{i'} < a_i$ for all $e_{i'} \in A_i$,
2. $a_{i'} < a_i \leq b_{i'}$ for all $e_{i'} \in B_i$, and
3. $a_i \leq a_{i'} \leq b_{i'} \leq b_i$ for all $e_{i'} \in C_i$.

Clearly, for every $j \leq |C_i|$ we have $A[i, j] = |e_i|$ since by definition of A, e_i is included in the solution, and adding any other $j - 1$ sets from C_i to the solution does not increase the size of the union. In the remainder of the proof, when we refer to an optimal solution corresponding to $A[i', j']$ for some indices i' and j' we always mean a solution that uses the maximum number of sets in $C_{i'}$.

For any $t \geq 0$ and $j = t + |C_i|$, the optimal solution contains exactly t sets in $A_i \cup B_i$. Fix an optimal solution OPT_i corresponding to $A[i, j]$ and let e_{i^*} be the hyperedge with largest $b_{e_{i^*}}$ in OPT_i that does not belong to C_i. We show that

$$A[i, j] = A[i^*, j - |C_i \setminus C_{i^*}|] + |e_i \setminus e_{i^*}|. \tag{2}$$

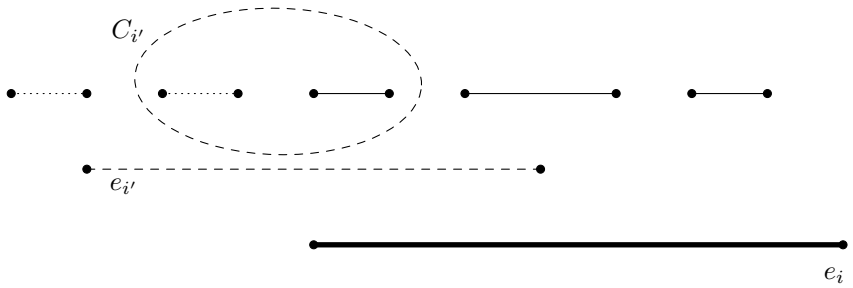


Figure 1 Partitioning of hyperedges induced by e_i. The dotted edges form set A_i, the dashed edge forms set B_i and the elements of C_i are represented by continuous edges. The set $C_{i'}$ is also shown in dashed pattern.

Then, by considering every hyperedge with index $i' < i$ as the possible i^* in Eq. (2) and taking the minimum value, one can compute $A[i, j]$ in linear time.

To complete the proof, we argue why Equation 2 holds. First observe that a solution with value $A[i, j]$ exists. Indeed, by adding all elements of $C_i \setminus C_{i^*}$ to an optimal solution for $A[i^*, j - |C_i \setminus C_{i^*}|]$ we obtain a solution for $A[i, j]$ covering exactly $|e_i \setminus e_{i^*}|$ additional elements. Next, assume that the value of $A[i, j]$ is less than that of Equation 2. Then we can obtain a solution for $A[i^*, j - |C_i \setminus C_{i^*}|]$ by removing from OPT_i all the elements in $|C_i \setminus C_{i^*}|$ to obtain a solution with value at most $A[i, j] - |e_i \setminus e_{i^*}|$, contradicting the fact that $A[i^*, j - |C_i \setminus C_{i^*}|]$ is the value of an optimal solution. ◀

8 Open problems

While no tight hardness results are known for Densest k-Subgraph and Smallest p-Edge Subgraph, there are lower bounds given by the log-density framework [7, 9]. In this framework, one considers the problem of distinguishing between a random graph and a graph which contains a planted dense subgraph. It has been conjectured that for certain parameters (namely, when the "log-density" of the subgraph is smaller than that of the host graph), this task is impossible, thus giving lower bounds on the approximability of these problems. In the graph setting, the existing algorithm of [7, 9] match these lower bounds.

However, in the hypergraph case, our current algorithms are still far from the corresponding lower bounds. In c-uniform hypergraphs, the lower bounds predicted by the log-density framework are $n^{(c-1)/4}$ for Densest k-Subhypergraph and $n^{1-2/(\sqrt{c}+1)}$ for Min p-Union. For $c = 3$, for example, these lower bounds give $n^{1/2}$ and $n^{2-\sqrt{3}} = n^{0.2679\ldots}$, respectively (contrast with our current guarantees of $n^{0.6978\ldots}$ and $n^{0.4}$). The existing approach for the graph case does not seem to easily carry over to hypergraphs, and it remains a technical challenge to match the log-density based predictions for hypergraphs of bounded rank.

For arbitrary rank, the lower bound given by the log-density framework is $m^{1/4}$ (note that we do not expect to achieve approximations that are sublinear in n in this case), as opposed to our current guarantee of $\sqrt{m}$. In general hypergraphs, one may also hope for hardness results which at the moment are elusive for the graph case or for bounded rank hypergraphs.

There is also an interesting connection between MpU/DkSH and the *Small-Set Vertex Expansion* problem (SSVE) [5, 20, 19]. In Small-Set Vertex Expansion we are given a graph

G and a parameter δ, and are asked to find the a set $V' \subseteq V$ with $|V'| \leq \delta n$ in order to minimize $\frac{|\{v \in V \setminus V' : v \in \Gamma(v)\}|}{|V'|}$. Given a graph G, consider the collection of neighborhoods $\hat{E} = \{\Gamma(v) : v \in V\}$ and the hypergraph $H = (V, \hat{E})$. If we let $p = \delta n$, the MpU problem (choosing p hyperedges in H to minimize their union) is quite similar to the SSVE problem. The main difference is that SSVE only "counts" nodes that are in $V \setminus V'$, while MpU would also count nodes in V'. It is known [21] that this special case of MpU reduces to SSVE, so it is no harder than SSVE, but it is not clear how much easier it is. This motivates the study of MpU when hyperedges are neighborhoods in an underlying graph, and studying the approximability of this problem is an interesting future direction.

References

1. Noga Alon, Sanjeev Arora, Rajsekar Manokaran, Dana Moshkovitz, and Omri Weinstein. Inapproximability of densest k-subgraph from average case hardness. Unpublished manuscript, 2011.
2. Benny Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *SIAM Journal on Computing*, 42(5):2008–2037, 2013. doi:10.1137/120884857.
3. Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Proceedings of the Forty-second ACM Symposium on Theory of Computing*, STOC'10, pages 171–180, 2010.
4. S. Arora, B. Barak, M. Brunnermeier, and R. Ge. Complicational complexity and information asymmetry in finnancial products. Submitted, 2016.
5. Sanjeev Arora and Rong Ge. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques: 14th International Workshop, APPROX 2011, and 15th International Workshop, RANDOM 2011, Princeton, NJ, USA, August 17-19, 2011. Proceedings*, chapter New Tools for Graph Coloring, pages 1–12. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-22935-0_1.
6. Aditya Bhaskara. *Finding Dense Structures in Graphs and Matrices*. PhD thesis, Princeton University, 2012.
7. Aditya Bhaskara, Moses Charikar, Eden Chlamtac, Uriel Feige, and Aravindan Vijayaraghavan. Detecting high log-densities: an $O(n^{1/4})$ approximation for densest k-subgraph. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 201–210, 2010.
8. Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In *Approximation Algorithms for Combinatorial Optimization, Third International Workshop, APPROX 2000, Saarbrücken, Germany, September 5-8, 2000, Proceedings*, pages 84–95, 2000. doi:10.1007/3-540-44436-X_10.
9. Eden Chlamtac, Michael Dinitz, and Robert Krauthgamer. Everywhere-sparse spanners via dense subgraphs. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 758–767, 2012.
10. Julia Chuzhoy, Yury Makarychev, Aravindan Vijayaraghavan, and Yuan Zhou. Approximation algorithms and hardness of the k-route cut problem. *ACM Trans. Algorithms*, 12(1):2:1–2:40, December 2015. doi:10.1145/2644814.
11. Michael Dinitz, Guy Kortsarz, and Zeev Nutov. Improved Approximation Algorithm for Steiner k-Forest with Nearly Uniform Weights. In Klaus Jansen, José D. P. Rolim, Nikhil R. Devanur, and Cristopher Moore, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*, volume 28 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 115–127, Dagstuhl, Germany, 2014. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: http://drops.

dagstuhl.de/opus/volltexte/2014/4692, doi:10.4230/LIPIcs.APPROX-RANDOM.2014.115.

12 Uriel Feige. Relations between average case complexity and approximation complexity. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC'02, pages 534–543, New York, NY, USA, 2002. ACM. doi:10.1145/509907.509985.

13 Uriel Feige, Guy Kortsarz, and David Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

14 Ashish Goel, Michael Kapralov, and Sanjeev Khanna. On the communication and streaming complexity of maximum bipartite matching. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 468–485, 2012.

15 Anupam Gupta, Mohammadtaghi Hajiaghayi, Viswanath Nagarajan, and R. Ravi. Dial a ride from k-forest. *ACM Trans. Algorithms*, 6(2):41:1–41:21, April 2010. doi:10.1145/1721837.1721857.

16 Subhash Khot. Ruling out ptas for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM Journal on Computing*, 36(4):1025–1071, 2006. doi:10.1137/S0097539705447037.

17 Christian Konrad and Adi Rosén. Approximating semi-matchings in streaming and in two-party communication. In *Automata, Languages, and Programming – 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part I*, pages 637–649, 2013.

18 Guy Kortsarz and David Peleg. On choosing a dense subgraph. In *34th Annual Symposium on Foundations of Computer Science, Palo Alto, California, USA, 3-5 November 1993*, pages 692–701, 1993.

19 Anand Louis and Yury Makarychev. Approximation Algorithms for Hypergraph Small Set Expansion and Small Set Vertex Expansion. In Klaus Jansen, José D. P. Rolim, Nikhil R. Devanur, and Cristopher Moore, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*, volume 28 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 339–355, Dagstuhl, Germany, 2014. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. URL: http://drops.dagstuhl.de/opus/volltexte/2014/4707, doi:10.4230/LIPIcs.APPROX-RANDOM.2014.339.

20 Anand Louis, Prasad Raghavendra, and Santosh Vempala. The complexity of approximating vertex expansion. *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, 0:360–369, 2013. doi:10.1109/FOCS.2013.46.

21 Yuru Makarychev. Personal communication.

22 Zeev Nutov. Approximating steiner networks with node-weights. *SIAM Journal on Computing*, 39(7):3001–3022, 2010. doi:10.1137/080729645.

A Finding a Set of Minimum Expansion

Given a bipartite graph $G = (E, V, F)$, the subroutine Min-Exp(G) returns a subset of E so that

$$\frac{|\text{Min-Exp}(G)|}{|\Gamma_G(\text{Min-Exp}(G))|} \geq \frac{|E'|}{|\Gamma_G(E')|},$$

for every subset $E' \subseteq E$. Minimally expanding subsets of this kind have previously been used (e.g. in [17, 14]) in communication settings where computation time is disregarded. We therefore present a polynomial time implementation for Min-Exp using network flows. An alternative algorithm can be derived from a straightforward adaptation of a linear

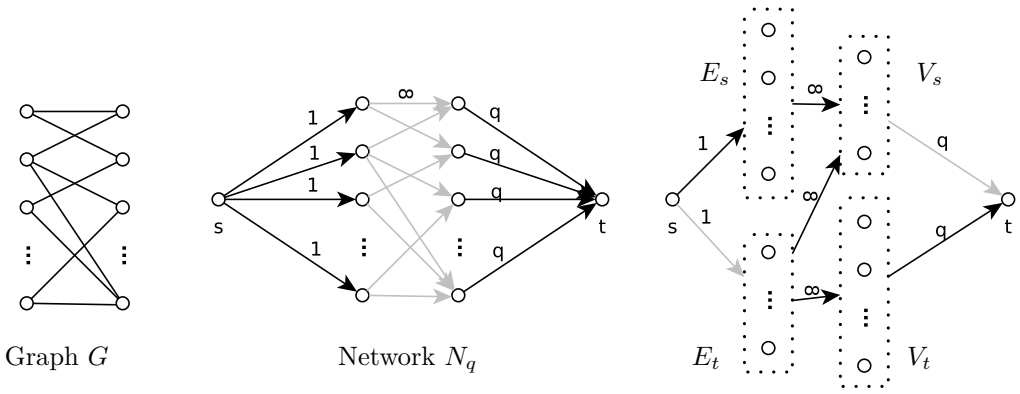


Figure 2 Left: Input graph G. Center: Network N_q. Right: Min-s-t-cut. Gray edges are cut edges.

programming approach for the graph case due to Charikar [8] to our setting (see Appendix B for more details).

Let $N_q = (\tilde{G}, c_q, s, t)$ be a flow network with directed bipartite graph $\tilde{G} = (E \cup \{t\}, V \cup \{s\}, \tilde{F})$, capacities c_q parameterized by a parameter q with $\frac{m}{n} < q < m$, source s and sink t as follows (and as illustrated in Figure 2):

1. Vertex s is connected to every $e \in E$ via directed edges (leaving s) with capacity 1.
2. Every $v \in V$ is connected to t via a directed edge (directed towards t) with capacity q.
3. Edges from F are included in $\tilde{F}$ and directed from E-vertex to V-vertex with capacity ∞.

Denote by C^* a minimum s-t cut in N_q and let $val(C^*)$ be the value of the cut. Since cutting all edges incident to vertex s results in a cut of value m, the min-cut value is at most m and thus finite, and, in particular, no edge connecting E to V is included in the min-cut. Denote by E_s the set of E-vertices that, when removing the cut-edges from the graph, are incident to s, and let $E_t = E \setminus E_s$. Let $V_s = \Gamma_G(E_s)$ and let $V_t = V \setminus V_s$. Since removing C^* from $\tilde{G}$ separates s from t, all outgoing edges from V_s are included in C^*. Furthermore, since C^* is a minimum cut, none of the edges leaving V_t are contained in the cut. The resulting structure is illustrated on the right in Figure 2. The value of the cut is computed as follows:

$$val(F^*) = |E_t| + q \cdot |V_s|. \tag{3}$$

We prove now a property connecting the value of a minimum cut to the expansion of a subset of E. This property allows us then to define an efficient algorithm for Min-Exp.

▶ **Lemma 24.** *Let q be such that $\frac{m}{n} < q < m$. Then:*

$$val(F^*) < m \Leftrightarrow \exists E' \subseteq E : \frac{|E'|}{|\Gamma_G(E')|} > q.$$

Proof. Suppose that $val(F^*) < m$. We prove that $E' = E_s$ fulfills the claimed property. The value of the cut $val(F^*)$ is computed according to Inequality 3 as follows:

$$m > val(F^*) = |E_t| + q \cdot |V_s| = m - |E_s| + q \cdot |V_s| = m - |E'| + q \cdot |\Gamma_G(E')|,$$

which implies $\frac{|E'|}{|\Gamma_G(E')|} > q$ as desired.

Suppose now that there is a $E' \subseteq E$ such that $\frac{|E'|}{|\Gamma_G(E')|} > q$. Then the set of edges C consisting of those that connect s to $E \setminus E'$ and those that connect $\Gamma_G(E')$ to t form a cut. We compute $val(C)$:

$$val(C) \quad = \quad |E \setminus E'| + q|\Gamma_G(E')| = m - |E'| + q|\Gamma_G(E')| < m - |E'| + |E'| = m.$$

The fact that $val(C^*) \leq val(C)$ completes the proof. ◀

Lemma 24 allows us to test whether there is a subset $E' \subseteq E$ such that $\frac{|E'|}{|\Gamma_G(E')|} > q$, for some value of q. For every set $E' \subseteq E$, we have $\frac{|E'|}{|\Gamma_G(E')|} \in \{\frac{a}{b} : a \in \{1, \ldots, m\}, b \in \{1, \ldots, n\}\}$. We could thus test all values $\frac{a}{b} - \epsilon$, for $a \in \{1, \ldots, m\}, b \in \{1, \ldots, n\}$ and a small enough ϵ, in order to identify the desired set (or use a binary search to speed up the process). Since computing a min-cut can be done in polynomial time, we obtain the following theorem:

▶ **Theorem 25.** *Algorithm* Min-Exp *can be implemented in polynomial time.*

B An LP-based algorithm for Minimum Expansion

We use hypergraph notation in this section. So the goal is to find a set $E' \subseteq E$ which minimizes $|\cup_{e \in E'} e|/|E'|$ over all choices of E' (so there is no requirement that $|E'| = p$).

We use the following LP relaxation, which is a straightforward adaptation of Charikar's [8] algorithm for graphs.

$$\begin{aligned}
\text{LP} = \min \quad & \sum_{i \in V} x_i & & \\
\text{s.t.} \quad & \sum_{e \in E} y_e = 1 & & \\
& x_i \geq y_e & & \forall e \in E, i \in e \\
& x_i \geq 0 & & \forall i \in V \\
& y_e \geq 0 & & \forall e \in E
\end{aligned}$$

Consider the following simple rounding algorithm:

- Pick $r \in_R [0, 1]$ uniformly at random.
- Let $E' = \{e \in E \mid x_e \geq r\}$.
- Let $V' = \bigcup_{e \in E'} e$.

Clearly, for every vertex $e \in E$ we have

$$\text{Prob}[e \in E'] = y_e.$$

Also, for every vertex $i \in V$ we have

$$\text{Prob}[i \in V'] = \max_{e \ni i} y_e \leq x_i.$$

Therefore, by linearity of expectation, we have

$$\mathbb{E}[\text{LP} \cdot |E'| - |V'|] \geq \text{LP} \cdot 1 - \text{LP} = 0,$$

and this is obviously still true when we condition the expectation on $|E'| > 0$ (a positive probability event), so with positive probability, we get a pair (V_0, E_0) such that $E_0 \neq \emptyset$, $V_0 = \bigcup_{e \in E_0} e$ and $|V_0|/|E_0| \leq \text{LP}$. The rounding is trivially derandomized by trying $r = y_e$ for every vertex $e \in E$.